Victoria University of Wellington
School of Engineering and Computer Science

SWEN222: Software Design, Assignment 1

Due: Friday 5th August @ Mid-night

1 Introduction

You are to implement parts of a program allowing people to play the game of Cluedo on a computer. If you have not heard of the game Cluedo before, don't worry a specification detailing what you need to do is provided. For the purposes of this assignment, your program will be a working prototype and will run on an ordinary desktop computer.

You will be given a rough specification of the Cluedo game, and asked to design and implement a solution. Your program will be implemented in Java and structured using classes, inheritance, and collections.

2 Specification

What follows is a simple specification for the game of Cluedo. The rules of the game have been simplified for the purposes of this assignment. Furthermore, the specification is neither complete nor extremely detailed and you must make reasonable assumptions where necessary.

2.1 Objective

The game consists of between three and six players who move around a board consisting of different rooms. The aim is to deduce who the murderer was, what weapon they used and what room it happened in.

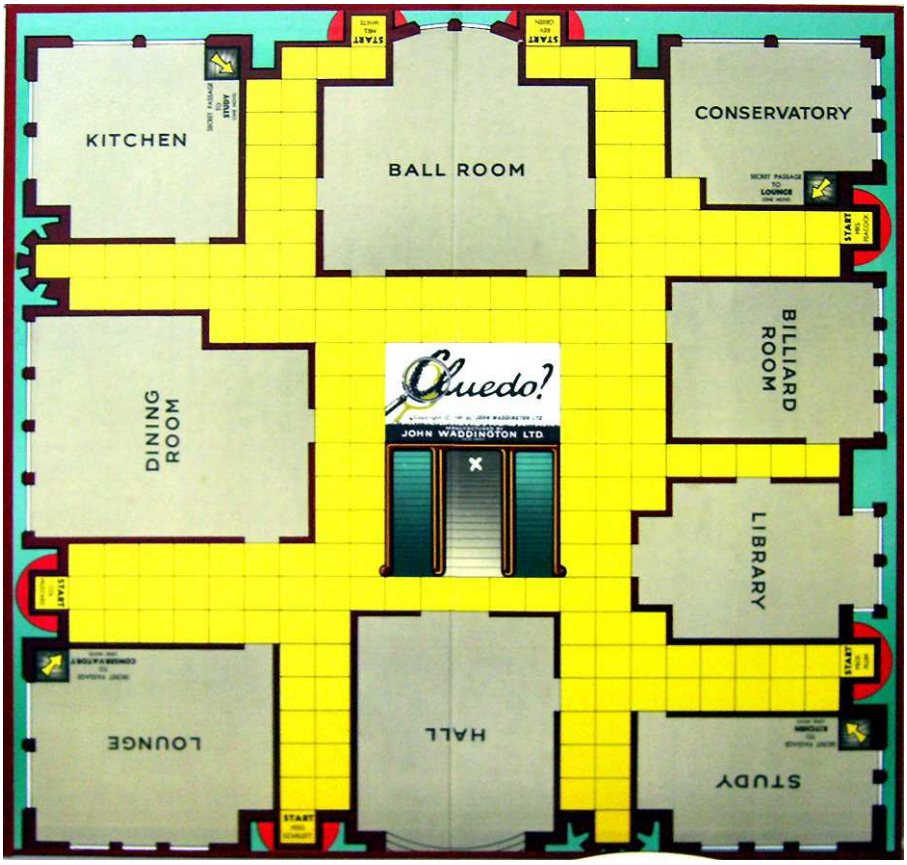

Figure 1: The Cluedo Board

2.2 Board

The Cluedo board consists of nine rooms laid out in a circular fashion (see Figure 1). The board is divided into a grid of (approx) 25x25 squares. The corner rooms each contain a stairwell which connects to the room in the opposite corner of the board. The center-most room is not used during play, and holds the solution envelope. There are six starting squares located on the outside of the board. Each starting square indicates which character starts at that position.

2.3 Character

There are six characters in the game, one of which is the murderer:

- Miss Scarlett
- Colonel Mustard
- Mrs. White
- The Reverend Green
- Mrs. Peacock
- Professor Plum

Each player in the game assumes the role of one of these characters. There may be unused characters if there are less than six players.

2.4 Weapons

There are six weapons in the game, one of which is the murder weapon:

- Candlestick
- Dagger
- Lead Pipe
- Revolver
- Rope
- Spanner

Each weapon is located in a room chosen at random, such that no two weapons are in the same room.

2.5 Rules

Every character, weapon and room is represented by a special card in the game. Before the game starts one character, one weapon and one room card are selected at random and placed in a special envelope. This is then located in the center of the board. These cards represent the "solution" — that is, they identify the murderer, the murder weapon and murder room. The remaining weapon, room and character cards are then dealt evenly to the players.

Players take it in turns to roll the dice and move their character token a corresponding number of squares. When a player enters a room he/she can announce a *suggestion* which consists of that room, and a character and weapon (if necessary, the character and weapon pieces are moved into the room). By making the announcement, the player is hypothesising that these solve the murder.

When an announcement is made, each player responds in a clock-wise fashion until either: the suggestion is refuted; or, all players have indicated they cannot refute the suggestion. A suggestion is refuted by a player if that player contains a matching card. In other words, the player contains a card and, hence, that card cannot be in the solution envelope.

Play continues until a player believes he/she has determined the solution. At that point, he/she makes an *accusation* consisting of a weapon, character and room (note: unlike suggestions, the player does not need to be in a room to make an accusation involving it). The accusation is then checked by the player against the solution. If it is correct the player wins, otherwise the player is eliminated from the game.

2.6 User Interface

You are asked to implement an object-oriented program for playing the Cluedo game. This should provide a simple, text-driven user interface and should at least do the following:

1. The program begins by asking how many players are required.
2. At the start of each turn, the program roles the dice, moves the player's token to the desired spot.
3. Once a player has moved, he/she is presented with the option of making an announcement or an accusation. Note, the program should prevent the player from breaking the rules (e.g. announcing a hypothesis for a room the player is not currently in).
4. The program then repeats steps 2-4 for the next player and so on.

All input/output for this should occur via `System.in` and `System.out`. **DO NOT USE A GRAPHICAL USER INTERFACE MARKS WILL BE DEDUCTED FOR DOING THIS!**

3 What to Do

For this assignment, you have the option of:

1. **Working on your own.** In such case, all work must be your own.
2. **Working in a pair.** In such case, you may work together with another partner on the project. You must register your intent to do this using the team signup system:

`http://ecs.victoria.ac.nz/cgi-bin/teamsignup`

NOTE: In marking the assignment, no distinction will be made between students who worked in pairs and those who did not. Therefore, there is a clear advantage for students who chose to work in pairs. This is because the workload will be shared between them. We wish to encourage you to work in pairs, as this will help develop good teamwork skills and better prepare you for the group project.

You should begin by formulating a rough design for your project. Begin by constructing *CRC cards*, following the technique outlined in lectures. Then, construct a preliminary *UML class diagram*.

Once you are happy with your initial design, you should begin the implementation. During this, you will most likely want to refine your design as problems are uncovered. Finally, you should complete the remaining documents as necessary (see below).

4 Submission

Your program code should be submitted electronically via the *online submission system*, linked from the course homepage. Your submitted code should be packaged into a jar file, including the source code (see the export-to-jar tutorial linked from the course homepage). Your program should include appropriate Javadoc comments, and provide a suite of JUnit test cases.

In addition to the source code, various pieces of design documentation are required. These should be submitted as either PDF or PNG files; other formats will not be accepted. The required documents are:

1. **CRC Cards** for the main classes in the system.
2. **Class Diagram** illustrating the main aspects of the class hierarchy. This should not be cluttered with unnecessary information such as, for example, `toString` methods.
3. **Sequence diagram** illustrating the sequence of events which occur when a player takes his/her turn.
4. A **one-page report** written in your own words giving an overview of the design for the cluedo program.

NOTE: The written report must be your own work. That means you cannot submit a report which is identical to that of your other team member(s). Furthermore, it is not acceptable to simply copy material from other sources, such as the internet or text books.

5 Assessment

This assignment will be marked as a letter grade (A+ ... E), based primarily on the following criteria:

5.1 Individual Report [10 marks]

For the individual report, marks will be awarded on an *individual basis* as follows:

- **Grammar and Spelling (4 marks)**. Reports should be free from typographical and related errors.
- **Design Discussion (6 marks)**. Marks will be awarded for how well the individual report conveys aspects of the program's design. Marks will also be awarded for how well-written and easy to follow it is.

5.2 Group Design [30 marks]

For the design documents, marks will be awarded on a *group basis* as follows:

- **CRC Cards (10 marks)**. Marks will be awarded for clarity and presentation of the cards, as well as correct use of the notation.
- **Class Diagram (10 marks)**. Marks will be awarded for correct use of the following: *inheritance*, *multiplicity*, *associations*, *classes* and *attributes*. Marks will also be awarded for *clarity*.
- **Sequence Diagram (10 marks)**. Marks will be awarded for clarity and presentation of the sequence, as well as correct use of the notation. Marks will also be awarded for how well the diagram conveys the prescribed event.

5.3 Group Software [40 marks]

For the submitted code, marks will be awarded on a *group basis* as follows:

- **Correctness (15 marks)**. Marks will be awarded for programs which correctly implement the given specification of Cluedo.
- **Execution (5 marks)**. Marks will be awarded for programs which execute correctly without crashing. This includes appropriate handling of invalid input (e.g. text entered when a number is expected).

- **Text Interface (5 marks)**. Marks will be awarded for the text interface which includes: ease of use and overall clarity of presentation.
- **Code Style (5 marks)**. Marks will be awarded for overall code style which includes: good use of naming for methods, fields, classes and variables method naming, field naming, classes and variables; good division of work into methods, so as to avoid long and complex chunks of code; consistent formatting (e.g. indentation, etc).
- **Commenting (5 marks)**. Marks will be awarded for good use of JavaDoc comments on all public classes, interfaces and methods. Marks will also be awarded for good use of internal (i.e. non-javadoc) comments. Typically, these are found within a method body describing some aspect of how it works. Comments should make sense and correctly describe what is happening.
- **JUnit (5 marks)**. Marks will be awarded for the inclusion of sensible JUnit tests. These should cover the basic functionality of the program and it is expected that *at least 20 test cases would be included*.